Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in the application:

Listing of Claims:

Claim 1 (Currently Amended): An application programming interface for a programmable graphics processor pipeline, comprising:

one or more program instructions to configure a fragment processor within the programmable graphics processor pipeline to detect a position conflict for an x, y position and prevent a subsequent access of the position until the position conflict is resolved, the instructions further including one or more instructions to write the x, y position upon completion of a first fragment process and read the x, y position to process a second fragment which depends on the first fragment process being completed without an intervening instruction to flush the graphics pipeline.

Claim 2 (Original): The application programming interface of claim 1, wherein a program instruction receives as input a source location and a destination location.

Claims 3-10 (Cancelled)

Claim 11 (Currently Amended): A fragment program for processing fragment data in a fragment processing pipeline, comprising a sequence of instructions comprising:

shading a first fragment associated with a destination location in a buffer;

~~a fragment program instruction to write~~ storing the result of the first fragment shading in the [[a]] destination location in [[a]] the buffer; [[and]]

~~a fragment program instruction to read the destination location in the buffer, without an intervening instruction to flush the fragment processing pipeline~~

shading a second fragment which is to be based in part on reading the result of the first fragment shading from the destination location;

detecting in a conflict detection unit that a read after write position conflict exists for the destination location in the buffer associated with the first and second fragments and interrupting the processing of second fragment; and

<u>on completing processing of the first fragment and storing the result in the destination location in the buffer resuming processing of the second fragment, processing of the second fragment including reading the destination location in the buffer without an intervening instruction to flush the fragment processing pipeline</u>.

Claim 12 (Original): The fragment program of claim 11, wherein the destination location includes a buffer identifier corresponding to one of several buffers.

Claim 13 (Original): The fragment program of claim 11 comprising fragment program instructions to configure the fragment processing pipeline to perform depth buffering prior to shading.

Claim 14 (Original): The fragment program of claim 11, comprising fragment program instructions to configure the fragment processing pipeline to perform depth peeling.

Claim 15 (Original): The fragment program of claim 11, comprising:

fragment program instructions to configure the fragment processing pipeline to perform raster operations.

Claim 16 (Previously Presented): The fragment program of claim 11, wherein raster operations are performed using fragment data represented in a floating-point data format.

Claims 17-22 (Cancelled)

Claim 23 (Previously Presented): A method for processing fragments in a graphics processor pipeline, comprising:

providing a fragment processing unit within the graphics processor pipeline;

receiving a first fragment associated with a position by the fragment processing unit;

processing the first fragment associated with the position to obtain a processed first fragment;

receiving a second fragment associated with the position by the fragment processing unit;

interlocking the second fragment in part subject to completion of the processing of the first fragment;

writing the processed first fragment to a graphics memory;

unlocking the second fragment; and

processing the second fragment in the fragment processing unit without flushing the pipeline between processing the first and second segments.

Claim 24 (Previously Presented): A method as claimed in claim 23, including processing one or more additional fragments following processing the first fragment without unlocking the second fragment.

Claim 25 (Previously Presented): A method as claimed in claim 23, including specifying the position of the first segment as source data for subsequent processing of fragments.

Claim 26 (Previously Presented): A method as claimed in claim 25, wherein the interlocking step comprises reaching the source data prior to processing the second fragment to prevent writing to the position.

Claim 27 (Previously Presented): The application programming interface of claim 1, wherein the position comprises a region including a plurality of pixels.

Claim 28 (Previously Presented): A method as claimed in claim 23, including checking a location in graphics memory for the processed first fragment prior to unlocking the second fragment.

Claim 29 (Previously Presented): A method as claimed in claim 23, including processing the first and either the second or the additional fragments in parallel.

Claim 30 (Previously Presented): A programmable graphics processor for execution of program instructions, comprising:

a read interface configured to read data from a graphics memory;

a fragment processing unit configured to receive fragments, each fragment associated with a position, and the data from the graphics memory and generate processed fragments;

a conflict detection unit configured to selectively store the position associated with each fragment and generate a position conflict status;

a write interface configured to write the processed fragments to the graphics memory; and

a fragment processing pipeline configured to handle read-after-write hazards during execution of shader programs including an instruction to write a location in graphics memory, an instruction to check the location in a conflict detection unit and a subsequent instruction to read a location in graphics memory without an intervening instruction to flush the fragment processing pipeline based on the check of the conflict detection unit.

Claim 31 (Previously Presented): A programmable graphics processor as claimed in claim 30, including a data cache storing additional data associated with the location, the conflict detection unit determining if the additional data associated with the location is available.

Claim 32 (Previously Presented): A programmable graphics processor as claimed in claim 31, wherein the location is a region comprising a plurality of pixels.

Claim 33 (New): A program as in claim 11, including:

outputting write position information to the conflict detection unit confirming that shading of the first fragment and writing to the buffer is complete.

Claim 34 (New): An application programming interface as claimed in claim 1, wherein the x, y position comprises coordinates within a display.

Claim 35 (New): A method as claimed in claim 11, including processing one or more additional fragments following processing the first fragment without unlocking the second fragment.